Energy saving and network performance: a trade-off approach

Energy Saving and Network Performance: a Trade-off Approach

Carla Panarello
Dipartimento di Ingegneria Informatica e delle Telecomunicazioni
Università di Catania - Italy
cpana@diit.unict.it

Alfio Lombardo
Dipartimento di Ingegneria Informatica e delle Telecomunicazioni
Università di Catania - Italy
alombard@diit.unict.it

Giovanni Schembra
Dipartimento di Ingegneria Informatica e delle Telecomunicazioni
Università di Catania - Italy
schembra@diit.unict.it

Luca Chiaraviglio
Dipartimento di Elettronica
Politecnico di Torino - Italy
luca.chiaraviglio@polito.it

Marco Mellia
Dipartimento di Elettronica
Politecnico di Torino - Italy
mellia@tlc.polito.it

ABSTRACT

Power consumption of the Information and Communication Technology sector (ICT) has recently become a key challenge. In particular, actions to improve energy-efficiency of Internet Service Providers (ISPs) are becoming imperative. To this purpose, in this paper we focus on reducing the power consumption of access nodes in an ISP network, by controlling the amount of service capacity each network device has to offer to meet the actual traffic demand. More specifically, we propose a *Green router* (G-router) implementing a congestion control technique named Active Window Management (AWM) coupled with a new capacity scaling algorithm named Energy Aware service Rate Tuner Handling (EARTH). The AWM characteristics allow to detect whether a waste of energy is playing out, whereas EARTH is aimed at invoking power management primitives at the hardware level to precisely control the current capacity of access nodes and consequently their power consumption. We test the benefits of the AWM-EARTH mechanism on a realistic scenario. Results show that the capacity scaling technique can save up to 70% of power consumption, while guaranteeing Quality of Service and traffic demand constraints.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; C.4 [**Performance of Systems**]: Design studies

General Terms

Design, Management, Performance

1. INTRODUCTION

The steadily rising energy cost and the need to reduce the global greenhouse (such as CO_2) gas emission to protect our environment have turned energy efficiency into one of the primary technological challenges of our century [1]. In this context, Information and Communication Technology (ICT) is expected to play a major active role in the reduction of the world-wide energy requirements, through the optimization of energy generation, transportation, and consumption. However, a number of studies estimate a power consumption related to ICT itself varying from 2% to 10% of the world-wide power consumption [2], [3]. This trend is expected to increase notably in the near future. Not surprisingly, only 20% of ICT carbon emissions derive from manufacturing, while 80% arise from equipment use [4]. Moreover, among the main ICT sectors, 37% of the total ICT emissions are due to the Telecom infrastructures and devices, while data centers and user terminals are responsible for the remaining part [4]. In Italy, for example, Telecom Italia is the second largest consumer of electricity after the National Railway system [5], consuming more than 2TWh per year. Similar and even more pessimistic considerations hold for the other developed countries (see for example [6] for the case of Japan).

To this extent, networking devices like backbone IP routers consume the large majority of energy [7], and the energy consumption of a network can almost double when including air conditioning and cooling costs. It is therefore not surprising that researchers, manufacturers and network providers are spending their effort to reduce the power consumption of ICT systems from different angles.

In the literature (see Sec. 2 for an overview), several proposals aim at reducing the overall network power consumption by turning off nodes and links, and re-routing traffic to save energy [8, 9, 10, 11]. However, these policies can be applied to transit nodes/links only, since at the network access, nodes and links cannot be turned off to avoid disconnecting users. However, notice that the largest fraction of power consumption of an Internet Service Provider (ISP) network is due to access nodes rather than core nodes. In this paper, we focus on reducing the power consumption of access nodes, while aiming at controlling traffic they carry,

by finding the minimum amount of capacity each network device has to offer to meet the actual traffic demand. The intuition is to allow network nodes to adapt the switching and transmission capacity to the current traffic demand to save energy and reduce power consumption. Indeed, as recently shown in [9], current nodes have a power consumption that is practically constant and independent from the actual load they face, causing a large waste of energy. We therefore support the adoption of both capacity and switching control mechanisms that would allow each node to meet the current traffic demand and save energy. We name this capability *Active Capacity Scaling* (ACS). Notice that, while current nodes do not offer support for capacity scaling, the technology to implement variable capacity electronic devices is readily available, as for example implemented in modern PCs and mobile devices in general.

Active Capacity Scaling capability requires the access node is able to estimate both the maximum capacity the user traffic requires, and the minimum available capacity in the network path between source and destination. Indeed the access node has to discover the minimum between the above capacity values, which, in fact, represents a resource allocation trade-off that allows the access node to provide the user with the maximum QoS while minimizing the energy consumption.

Estimating the above "trade-off capacity" is not a trivial task in current Internet. This is due to the elastic nature of traffic, and of TCP in particular, so that, if capacity is reduced/increased at the bottleneck link, the instantaneous traffic offered by TCP sources would reduce/increase consequently. The intuition therefore suggests that controlling the trade-off capacity is a multi-constrained problem that must be carefully studied. A similar problem has been studied in the literature to solve the Internet congestion control problem [12]; in this case the user traffic is driven by an implicit or explicit signaling which results from an accurate monitoring and control of the queue length in the network nodes ([13], [14], [15]). Similarly, we propose to estimate both the current offered traffic and the bottleneck capacity by monitoring and carefully controlling the queue length of transmission links in the access nodes, so that if the queue empties, the transmission capacity is reduced, while it is increased if the queue risks to overflow. For this purpose, we propose to use a congestion control technique named Active Window Management (AWM) ([16], [17]) coupled with an Energy Aware service Rate Tuner Handling (EARTH) mechanism aimed at invoking power management primitives at the hardware level to increase/decrease the node operational rate and the related performance states step by step.

More specifically, in this paper we introduce a *Green router* (G-router) implementing both the AWM and the EARTH algorithms. We show that AWM-EARTH mechanisms provide for ACS capability in the access nodes, therefore producing large energy saving to network operators. To support this statement, we present a reality-based analysis, derived considering the network topology of an actual ISP, and the daily traffic profile it has to carry. Results show that the energy saving is proportional to the capacity saving ACS can guarantee, so that during off-peak hours it tops to 70% of total access network power consumption, i.e., more than 45% of total network consumption. Let us note that the deployment of AWM-EARTH requires changes at access nodes only, which is a major advantage considering the current and future Internet ([16], [17]). Moreover, AWM is scalable with the amount of managed input traffic, and the proposed G-routers are compatible with the current routers today existing in Internet.

The paper is organized as follows. Sec. 2 overviews the related works. Sec. 3 presents the rationale of the ACS capability on a realistic scenario. Sec. 4 details the AWM mechanism and the EARTH algorithm, which provides for the ACS capability in the access nodes. Simulation results are presented in Sec. 5. Finally, conclusions are drawn in Sec. 6 .

2. RELATED WORK

The study of power-saving network devices has been introduced over these years, starting from the pioneering work of [8], in which for the first time authors face problem of quantifying and reducing the energy cost of the Internet. However, the attention of the research community and of Telecom operators only recently started to focus on this theme. Schemes to reduce the power consumption of links and nodes have then been proposed. In [18], Adaptive Link Rate (ALR) and protocol proxying are proposed, according to which a pair of high/low power and performance devices are present, being only one active a given time to match current demand. Both protocol and architectural changes are required to support this kind of approaches.

More recently, some effort was devoted to investigate how to reduce the power consumption of the entire network infrastructure, and not of single or few components only. In [9] some simple measurements about power consumption of networking devices are first presented; then authors consider a network topology and evaluate the total network consumption given the power footprint of each element. They consider two scenarios: in the first one, all devices are turned on, while in the second one only the minimum number of elements to guarantee the service are actually powered on. In [10, 11] we proposed more complex algorithms to face the problem of finding the minimum amount of resources that have to be powered on to match the current traffic demand. In [11] a real test case similar to the one proposed here has been studied. Finally, in [19] the authors exploits the idea of exchanging energy profiles among devices to reduce the overall power consumption during routing and traffic-engineering operation. The impact of using different energy profiles is assessed on a realistic core topology, showing that a routing energy-aware can reduce the overall energy consumption.

The industries as well have recognized the importance of energy savings as a source of profit. Cisco for example is developing new solutions [20] to control and reduce the power consumption for all networked devices in enterprise networks, proposing a centralized management. Novel devices that allow to enter different power states are expected in the near future.

The evaluation of power-aware management schemes for networks is argued in [21]. In particular, the authors evaluate the energy savings from sleeping, i.e. during absence of packets, and the possible savings from rate-adaption. In this paper we propose an actual method to achieve rate-adaptation while at the same time improving network performance. Similarly, considering the single devices, energy aware solutions involving switches and software routers have been proposed by [22]. Authors propose three schemes for

Table 1: Power Consumption of Nodes

Node Type	Power [kW]	Fraction of Total Node Power
Core	10	9.46%
Backbone	3	19.03%
Metro	1	6.32%
Access nodes	2	65.19%

power reduction in network switches which are adaptive to changing traffic patterns and automatically tune their parameters to guarantee a bounded and specified increase in latency. Targeting enterprise switches, and proposing a novel architecture for buffering ingress packets using shadow ports, their schemes produce power savings of 20 to 35%. Authors in [23] provide similar performance figures, by presenting actual measurements on software routers that exploit energy features of PCs which can reduce the processing power during under loaded conditions. In our work, we go a step further, by proposing a combined congestion control and rate-adaptation scheme for Internet access nodes which allows to both reduce power consumption and to better control traffic offered by the users.

3. GREEN NETWORKING: SCENARIO AND RATIONALES

In this section, we provide real data sets to support the need for power-adaptation schemes to save energy. We consider a topology, a peak-hour traffic demand, and a traffic pattern that varies following the typical day/night pattern. We then compute the energy required to run the network at full capacity, and the saving that the ACS scheme would provide. The topology, the traffic demand, its daily pattern and the power consumption of nodes and links are derived from measurements on a real ISP topology in Italy, so that the considered scenario is representative of the one faced by a national ISP. In the following, we detail first the considered scenario, presenting the topology, the traffic pattern and node power requirements. We then compute the energy consumption to compare the eventual energy saving the ACS scheme can provide.

3.1 Physical Topology

The topology considered in this work is similar to the actual topology of an Italian ISP. It follows a hierarchical design, as reported in Fig. 1, in which four levels of nodes are present: core, backbone, metro and access nodes. The inner level is composed by "core nodes" (Fig. 2.a), that are densely interconnected by 70 Gbps links. Core nodes are placed in four central Points-of-Presence (POPs) located in two cities. Each central POP hosts a pair of core nodes, each connected to other core nodes by two links for failure protection. A high-capacity Internet peering router is connected to two central POPs to offer connectivity to the Internet by means of four 70 Gbps links.

At the second level, so called "backbone nodes" (Fig. 2.b) are connected to the core by 8.5 Gbps links. Each backbone node is connected to two central POPs. Backbone nodes are located in large POPs, named "chief POPs", spread in each large city. At the third level, "metro nodes" (Fig. 2.c) are present. Each metro node is dual-homed to two backbone nodes by 8.5 Gbps capacity links. Metro and backbone nodes are located in the same chief POP[1]. The last level of nodes is represented by the "access nodes" (Fig. 2.d), that bring connectivity to the Digital Subscriber Line Access Multiplexers (DSLAMs) to which users are connected. Access nodes aggregate traffic from users in the same neighborhood or small town. Each access node is dual-homed to the closest pair of metro nodes by 0.75 Gbps capacity links.

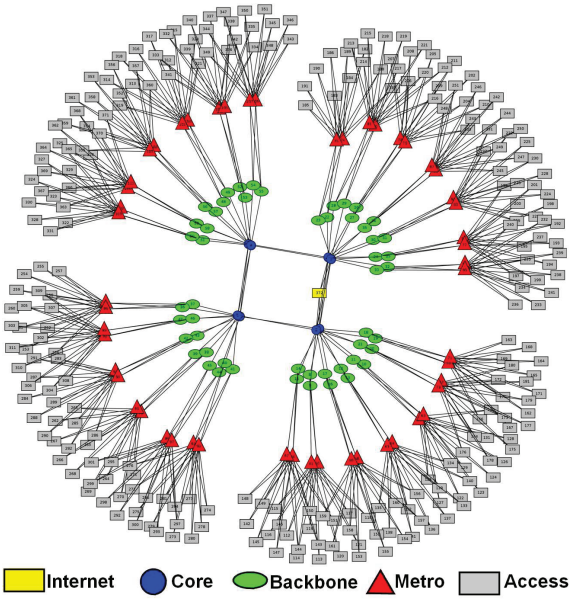

Figure 1: Schematic representation of the considered topology.

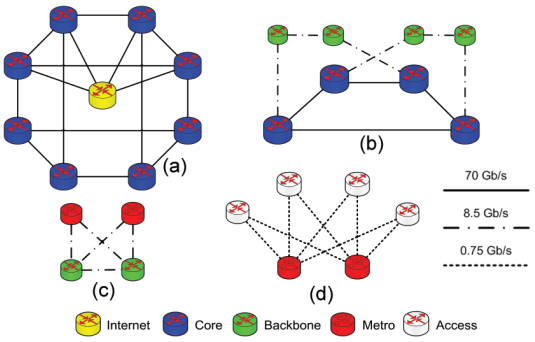

Figure 2: Link description

In this paper we consider a network composed by 372 routers: 8 core nodes, 52 backbone nodes, 52 metro nodes and 260 access nodes. 718 links in total are present.

To model the energy consumption of routers and links, we consider the specifications of real devices, as provided by the router manufacturers. Table 1 reports the mean power consumption for the different classes of nodes[2]. Each transmission line card is assumed to consume 100 Wh, as in [9].

[1]Notice that chief POPs are composed also by other elements, e.g the Network Access Servers (NASs) that allows user authentication. These devices are not considered in this work.

[2]These values do not consider air conditioning costs, which can be of the same order of magnitude of device power consumption.

The total power consumption of network nodes and links amounts to about 1 MW. As expected the large majority of nodes are access nodes, which supports the need for ACS to reduce their power consumption and waste.

3.2 Traffic Demand

We now model the peak-hour traffic demand the user offer to the network. We consider the actual traffic pattern as provided by the operator. To derive then the traffic volume each node pair exchange, we compute the maximum amount of traffic the network can support given an overprovisioning factor η =0.5, which is typically used by network operators. The access and the Internet peering nodes are the only possible sources and destinations of traffic.

Traffic estimates of the considered ISP show that during peak-hour about 70% of the total traffic amount is exchanged between the Internet at large and the ISP users, while the remaining part is exchanged uniformly among the ISP access nodes, i.e., 30% of traffic is confined within the same ISP, while 70% of traffic is coming from and going to other ISPs. Given $N+1$ nodes, let g^{sd} be the average amount of traffic from node $s = 0, \ldots, N$ to node $d = 0, \ldots, N$, i.e., $\{g^{sd}\}$ is the "peak-hour traffic matrix". Let $g^{s\cdot} = \sum_d g^{sd}$ ($g^{\cdot d} = \sum_s g^{sd}$) the total traffic generated (received) by node $s(d)$. Let node $i = 0$ be the Internet peering node. Then we have:

$$\begin{cases} g^{s0} = 0.7g^{s\cdot} & \forall s \\ g^{0d} = 0.7g^{\cdot d} & \forall d \\ \sum_{d=1}^{N} g^{sd} = 0.3g^{s\cdot} & \forall s \\ \sum_{s=1}^{N} g^{sd} = 0.3g^{\cdot d} & \forall d \end{cases} \quad (1)$$

To generate a possible traffic matrix, we assume that each link utilization cannot grow above 50% of the link capacity as overprovisioning constraint ($\eta = 0.5$), so that:

$$f^{ij} \leq \eta C^{ij} \qquad \forall i, j \quad (2)$$

where f^{ij} is the total amount of traffic flowing on link from i to j during peak-hour, and C^{ij} is the link capacity. For simplicity, we assume that g^{sd} are i.i.d. random variables, distributed according to a uniform distribution, i.e., $E[g^{s0}] = E[g^{0d}] = 0.7$ units of traffic, and $E[g^{sd}] = 0.3/N$ units of traffic, and $std[g^{sd}] = E[g^{sd}]$.

The algorithm used to derive a possible traffic matrix works as follows: we generate a random traffic matrix $\{\hat{g}^{sd}\}$; then we route the traffic in the network according to a minimum hop path routing. In case of tie, a random path is selected among the minimum hop paths to exploit network redundancy to balance the link load. We then compute the total amount of traffic flowing on each link, and look for the mostly loaded link $(i,j)^* = \text{argmax}(\max_{(i,j)} f^{ij}/C^{ij})$. We then define a scaling factor $\alpha = \eta \frac{C^{ij}}{f^{ij}}$, $(i,j) = (i,j)^*$, and compute the peak-hour traffic demand as

$$g^{sd} = \alpha \hat{g}^{sd} \quad (3)$$

This guarantees that the constraint of Eq.(2) holds true for all links and that there is at least one link whose offered load is equal to the overload bound.

3.3 Performance Evaluation

We consider a scenario in which traffic varies according to the day-night pattern observed in the real network. Top plot of Fig. 3 reports the total offered traffic defined as $\mathcal{G}(t) = \sum_{s,d} g^{s,d}(t)$. A time period of 24 hours corresponding to a typical working day is shown. Values are averaged over 5 minutes, and a normalization factor has been applied due to non-disclosure agreements with the operator. For simplicity, we assume the same traffic pattern is affecting each traffic demand, so that it can be expressed as:

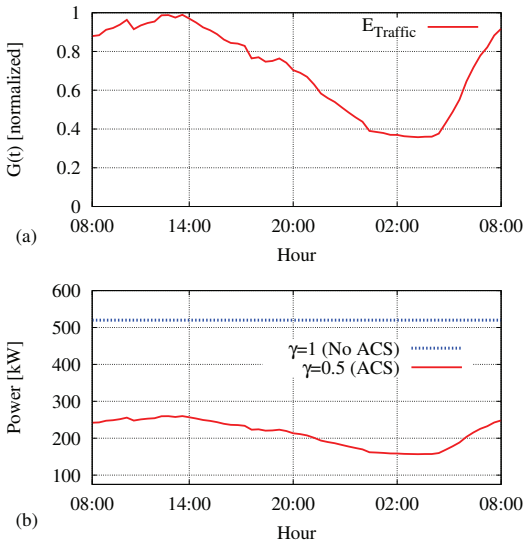

Figure 3: Total traffic daily pattern (top plot), and corresponding power variation when ACS is adopted (bottom plot).

$$g^{sd}(t) = a(t)g^{sd} \quad (4)$$

being $a(t) = \frac{\mathcal{G}(t)}{\max(\mathcal{G}(t))} \in [0:1]$ the shaping function at time t and g^{sd} the traffic exchanged by s and d during the peak hour.

To analyze the worst case scenario, we assume that only the access nodes implement the ACS policy, that is, they are the only nodes able to adapt the power consumption with the load according to the proposed policy.

Given that today power consumption of nodes is practically constant with load, researchers have modeled power consumption simply with "on-off" model [9]. Future devices instead will be able to scale their power with the load, adopting an energy-proportional paradigm [24], where power is linear with the throughput. In our work we choose the linear model for power consumption since we target the design of next-generation devices. In particular, the ACS node power consumption is modeled as a constant term P^C plus an additional term P^D that takes into account the impact of load variation. In other words, for a generic node i, we have:

$$P_i^{TOT} = \gamma P_i^D \left(\sum_j \frac{f^{ij}}{\eta C^{ij}} \right) + (1-\gamma) P_i^C \quad (5)$$

$\gamma \in [0,1]$ allows to model different power saving capabilities. For example, for $\gamma = 0$, P_i^{TOT} is constant, while for $\gamma = 1$, the node power consumption would be directly proportional to the node load. We set $P^D = 1600$ W and $P^C = 400$ W, to reflect the impact of traffic on power consumption, assuming P^D maximal when the amount of flow is equal to the link capacity, scaled by the overprovisioning constraint.

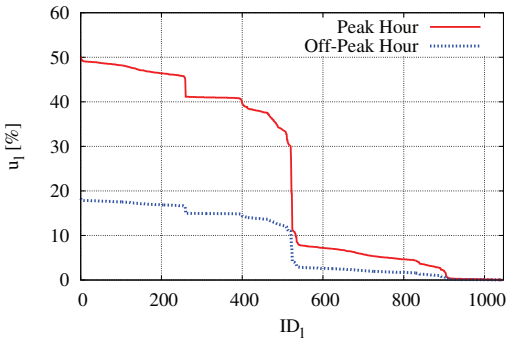

Figure 4: Access Link Utilization during Peak and Off-Peak Hours

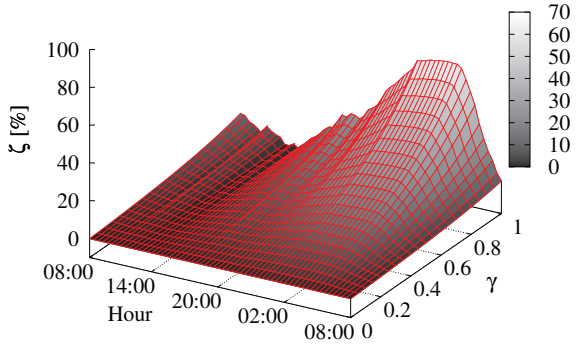

Figure 5: Power Saving variation versus time and γ.

3.4 Results

The bottom plot of Fig. 3 shows the total power consumed by the access nodes versus the time of the day. We set $\gamma = 0.5$ for simplicity. For completeness, we report the power consumption of today access network, computed assuming no ACS is present, i.e., $\gamma = 0$. Two considerations holds: first, the ACS guarantee a power saving proportional to the overprovisioning constraint (50% in our case), which forces traffic to not exceed ηC_{ij}. Second, the energy saving strictly follows the trend of $\mathcal{G}(t)$, since the ACS nodes are able to adapt their power to the current load. The peak power is reduced to 260 kW, while during the off peak-time the total access node power consumption is less than 156 kW, guaranteeing a reduction of more than 70%.

Fig. 4 details the utilization of links between access and backbone routers, considering the peak and the off-peak hours. Links are sorted into decreasing link utilization for easy of visualization. Interestingly, during the peak hour about half of the links are utilized for more than 30% (with some of them close the overbooking constraint). This is due to the dual homing policy, according to which two links connect one access node to two backbone nodes. Due to single path routing, traffic is preferentially routed through only one link, the second being used for failure protection, resulting in an unbalanced traffic. During off-peak time, all links are lightly utilized, never exceeding 20% utilization, suggesting that great savings can be achieved by the ACS mechanism.

To give more insight, Fig. 5 reports the possible power savings for different values of γ: Power saving ζ is computed as the percentage of the difference from the peak-hour and current time power consumption. The condition $\gamma = 0$ reflects the situation in which no ACS is present, leading to a clear waste of power. Instead, as γ increases, the saving become larger, reaching around 70% during night time, when ACS guarantees maximum saving.

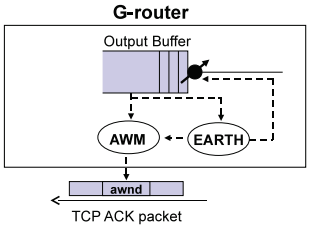

Figure 6: Scheme of the G-router.

4. CAPACITY SCALING TECHNIQUE

The analysis performed so far suggests that great savings can be achieved if the ACS mechanism is used. In this section we describe the Active Window Management (AWM) algorithm and the Energy-Aware service Rate Tuner Handling (EARTH), which provides for ACS capability in the access nodes. AWM is an Active Queue Management (AQM) mechanism the authors have introduced in [16] and [17] with the aim of making TCP transmissions with no loss, while maximizing network utilization. When AWM works as usual, the TCP source sending rate is driven by AWM in such a way that the packet arrival rate meets the service rate of an access bottleneck node and the output buffer queue length stabilizes around a given *target* value. Therefore, if the output buffer remains empty is just because the access router service rate is higher than the packet arrival rate. EARTH exploits this AWM characteristic to detect the minimum value of bandwidth necessary to guarantee the maximum allowed throughput without wasting energy. More specifically, by implementing the proposed mechanism on devices with capacity scaling capabilities, the EARTH algorithm may invoke power management primitives at the hardware level to decrease or increase the node operational rate and the related performance states. Fig. 6 show how AWM and EARTH cooperate. Let us refer to a node implementing the AWM-EARTH mechanism as *Green-router* (G-router). Let us note that the intrinsic AWM capability of stabilizing the queue length minimizes the number of hardware state transitions and the related hardware performance degradation. In the following we first shortly introduce the AWM mechanism and then we describe the EARTH algorithm.

4.1 The AWM Mechanism

The goal of AWM is to keep the output buffer queue length of an access bottleneck node close to the *target* value to achieve no loss, while maximizing network utilization. AWM is implemented on access nodes and interacts with TCP sources without requiring any modification to the TCP protocol. The AWM mechanism acts on ACK packets sent by receivers to the corresponding TCP sources. More specifically, it uses the TCP header field called *Advertised Window* (*awnd*) to control the transmission rate of the TCP sources. Let us recall that, according to TCP flow control, the TCP receiver uses *awnd* to inform the sender about the

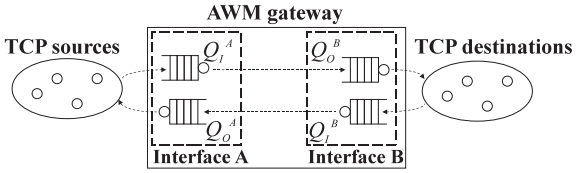

Figure 7: Buffers considered in the AWM algorithm.

number of bytes available in the receiver buffer. The TCP sender, on the other hand, transmits a number of packets given by the so-called *Transmission Window* (*twnd*), corresponding to the minimum value between its *Congestion Window* (*cwnd*) and the last received *awnd*. The basic idea of the AWM algorithm is to avoid losses by forcing the *awnd* value to be smaller than *cwnd* when the buffer queue length grows beyond a given *target* length, that is, when congestion is incoming but no loss has yet occurred. In this case, the AWM gateway modifies the *awnd* value in ACK packets to control the transmission rate of the TCP sources. By so doing, the *awnd* value received by the TCP sender may be lower than the one specified by the TCP receiver and the TCP AIMD mechanism is bypassed.

To avoid packet losses and maximize link utilization, the AWM algorithm calculates the value to be inserted in the *awnd* field in such a way that the average queue length in the gateway remains close to a *target* value, to be chosen significantly smaller than the buffer size (so as to have no losses), and at the same time higher than zero (to avoid under-utilization). To this end, based on the state of the queue, the AWM gateway estimates the number of bytes that it should receive from each TCP source. Hereafter we refer to this value as the suggested window (*swnd*).

In Fig. 7 we focus on two generic gateway interfaces, A and B. Let us note that, for each packet crossing an AWM gateway, the AWM algorithm considers both the output buffers $Q_O^{(B)}$, loaded by data packets coming from TCP sources, and $Q_O^{(A)}$, queuing the corresponding ACKs. Let us stress that the above applicability conditions occur in many relevant cases, for example in the access gateway of the majority of university campuses, factories, offices and home networks, that is, where small or large networks are connected to the Internet with a single router or an Internet Service Provider (ISP). The AWM gateway calculates the *swnd* value on the basis of the status of $Q_O^{(B)}$, and associates the network interface identifier B to this value. This is done at each packet arrival and departure in the $Q_O^{(B)}$ buffer, irrespective of the TCP connection the packet belongs to. Henceforward arrival and departure events in the $Q_O^{(B)}$ buffer will be called updating events. Let us now indicate the *swnd* value associated to the network interface B as $swnd^{(B)}$. Every time an ACK packet arrives in the input buffer $Q_I^{(B)}$, the network interface identifier B is associated to that ACK packet. Now let us assume that the ACK packet has to be forwarded through the interface A. When this ACK packet leaves the output buffer $Q_O^{(A)}$, its *awnd* value will be compared with the current $swnd^{(B)}$: if *awnd* is greater than the *swnd* value, the AWM gateway will overwrite the value of the *awnd* field with $swnd^{(B)}$, and recalculate the checksum; in the opposite case the *awnd* field value remains unchanged in order not to interfere with the original TCP flow control algorithm.

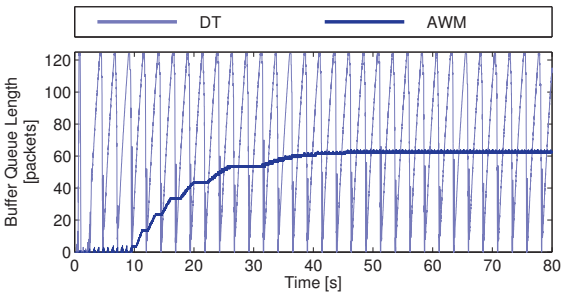

Figure 8: Buffer Queue Length comparison between AWM and DropTail (DT)

Let us stress that this mechanism is defined in such a way that scalability is not weakened. In fact, the AWM gateway maintains one *swnd* value for each network interface: no per-flow state memory is required, and updates are applied to *awnd* in ACKs coming from a given interface, irrespective of the particular TCP connection they belong to.

The *swnd* value at the generic updating event k, $swnd_k$, is evaluated on the basis of its previous value [3] $swnd_{k-1}$, by considering two corrective terms DQ_k and DT_k:

$$swnd_k = \max\left(swnd_{k-1} + DQ_k + DT_k, MTU\right) \qquad (6)$$

The minimum value of the *swnd* is set to the Maximum Transfer Unit (MTU) to avoid the *Silly Window Syndrome* [25].

The term DQ_k in (6) is defined as follows:

$$DQ_k = \left(q_{k-1} - q_k\right)/N_F \qquad (7)$$

where N_F is an estimate of the number of active TCP flows crossing the AWM gateway. In [16] the authors have defined an algorithm which allows to estimate it. The term DQ_k has been defined as in (7) to make a positive contribution when the instantaneous queue length q_k is lower than its previous value, and a negative contribution in the opposite case. In other words, the AWM gateway detects the bandwidth availability when the instantaneous queue length decreases, and informs the TCP sources by proportionally increasing the suggested window value for each of them.

When, on the contrary, the queue length increases, the AWM gateway infers incipient congestion and forces TCP sources to reduce their emission rates. If the N_F TCP sources reduce their transmission window by the term DQ_k, the queue length will go back to the q_{k-1} value.

The term DT_k, on the other hand, has been introduced to stabilize the queue length around a given *target* value, T. To this end, DT_k has to make a positive contribution when the queue length is less than T, and a negative contribution in the opposite case. For this reason we define:

$$DT_k = \frac{\beta \cdot (t_k - t_{k-1})}{N_F \cdot T} \cdot (T - q_k) \qquad (8)$$

where t_k and t_{k-1} are the time instants of the k-th and $(k-1)$-th updating events, respectively, and β is a positive parameter to be chosen to guarantee convergence to the *target*. In [16] we present an analytical fluid model of

[3] At the start-up time of the gateway, or after a long period during which the queue remains empty, AWM sets the initial value of *swnd* equal to one packet; this is done to drive the transmission rate of TCP sources immediately after the AWM gateway idle period.

the AWM gateway and a system stability study to be used to choose the appropriate values of β to obtain the desired performance.

As an example of the AWM behavior, Fig. 8 compares the evolution of the buffer queue length in an access node implementing AWM and in an access node with DropTail buffer policy[4]. As mentioned before, when AWM is implemented in the bottleneck node, the buffer queue length stabilizes around the *target* value assuring zero losses and high link utilization.

4.2 The EARTH algorithm

The mechanism presented so far is able to keep the output buffer queue length of the bottleneck node close to the *target* value. On the contrary, when the bottleneck is a node other than the G-router, the packet arrival rate in the G-router is lower than its service rate, and therefore its buffer empties. The same event occurs in the case the user traffic is lower than the node service rate. In both cases, to keep the queue length close to the *target* value, the AWM algorithm tries to increase the TCP source sending rate by indicating always increasing values of the suggested window. However, the TCP sources are not allowed to increase their sending rate because they are driven by either the forward and actual bottleneck node, or by the user traffic; so AWM indications do not take effect. This characteristic allows to reveal that the AWM service rate is higher than necessary, that is, it is higher than the minimum value of bandwidth necessary to guarantee the maximum allowed throughput without wasting energy.

With this in mind, in this section we propose a new algorithm, called Energy-Aware service Rate Tuner Handling (EARTH). By implementing it on device with capacity scaling capabilities, the EARTH algorithm may invoke power management primitives at the hardware level to decrease/increase the node operational rate and the related performance states step by step, until AWM regains the control of the TCP sending rate, that is until the AWM node service rate becomes the minimum between the source offered load and the forward bottleneck capacity.

More specifically, we distinguish two thresholds for the buffer queue length, *min_thresh* and *max_thresh*, such that:

$$min\_thresh < target < max\_thresh. \quad (9)$$

As soon as the buffer queue length decreases under the *target* value, the AWM algorithm indicates always increasing value of *swnd* to increase the TCP sources sending rate.

If the queue length does not approach again the target, on the contrary, if it becomes lower than *min_thresh* indicating that TCP sources are not capable to increase their sending rate, EARTH invokes the hardware management primitives to drive a one-step decrease of the node operational rate. After this occurrence, it may happen that the *swnd* value estimated by AWM during the previous empty buffer period is too high; for this reason, after any decrement of the node operational rate, the *swnd* value will be set to half of the one estimated before the operational rate decrease.

After any decrement of the operational rate, a time interval Δt_E is needed to allow AWM for estimating the new value of *swnd*. So the same actions will be repeated only if the queue length remains lower than *min_thresh* for the duration of the above time interval.

[4]Simulation results are obtained for the network topology in Fig. 10, when 10 greedy sources share a capacity of 10Mb/s. The round-trip propagation delay is 100ms and the buffer size is equal to bandwidth-delay product.

```
{State Update}
if (queue ≤ min_thresh) then
   state = Buff_EMPTY;
else if (queue ≥ max_thresh) then
   state = Buff_OVERFLOW;
else if (t − t_last_action ≥ Δt_PS) then
   state = PROBING;
end if

{State Action Setup}
if (state == Buff_EMPTY) then
   if (t − t_last_action ≥ Δt_E) or (last_state == PROBING)
   then
      action = DECREASE;
      swnd = last_swnd/2;
      last_swnd = swnd;
   else
      action = DO_NOTHING;
   end if
else if (state == Buff_OVERFLOW) then
   if (t − t_last_action ≥ Δt_O)
   or (last_state == Buff_EMPTY) then
      action = INCREASE;
   else
      action = DO_NOTHING;
   end if
else if (state == PROBING) then
   if (t − t_last_action ≥ Δt_P) or (last_state ≠ Buff_EMPTY)
   then
      action = INCREASE;
      last_swnd = swnd;
      swnd = swnd + pktsize;
   else
      action = DO_NOTHING;
   end if
end if

{State Action Execution}
if (action ≠ DO_NOTHING) then
   if (action == DECREASE) then
      C = C − ΔC;
   else if (action == INCREASE) then
      C = C + ΔC;
   end if
   t_last_action = t;
   last_state = state;
end if
```

Figure 9: Pseudo-code of the EARTH algorithm.

At the same way, when the buffer queue length exceeds the *target* value, the AWM algorithm decreases the *swnd* value. If the queue length, instead of decreasing, exceeds *max_thresh*, indicating that either the number of sources is higher than those AWM can manage[5] or a burst of traffic is occurring and more bandwidth is requested, EARTH invokes the hardware management primitives to step up the node service rate. Also in this case, a given time interval is needed to allow the system to react to the operational rate increase; so if immediately after the EARTH action, the buffer suddenly empties or the queue length still exceeds the *max_thresh* value, a new action will be considered only after Δt_E and Δt_O seconds, respectively.

[5]Let us remember that the AWM algorithm does not indicate values of *swnd* lower than 1 packet.

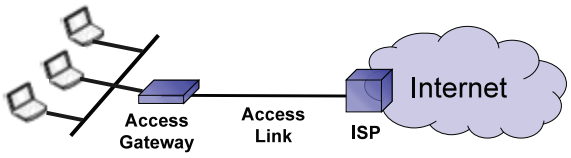

Figure 10: The simulated network topology: Access Gateway implements AWM-EARTH.

When the AWM algorithm is driving the TCP source sending rate, the node service rate is fully exploited, and the queue is close to the target. Unfortunately, in this case, the AWM mechanism does not allow to detect the occurrence in which either the sources are slowly increasing their offered load or a bottleneck node in the path has increased its available bandwidth. In both cases therefore the AWM node is forcing a bandwidth lower than is necessary, that is, a higher QoS could be provided to the users. To avoid this "QoS waste", each Δt_P seconds after the last change of the node operational rate, the EARTH algorithm invokes the hardware primitives to increase the node operational rate: if the buffer empties again, the algorithm immediately steps down the operational rate to the previous value; on the contrary, if the buffer queue length stabilizes again around the *target* value, a new operational rate increasing attempt is done after a time interval $\Delta t_{P_S} < \Delta t_P$.

In Fig. 9, the pseudo-code resumes the procedure executed by the described mechanism at the generic time instant t.

The technique proposed so far uses the AWM algorithm as a detector that a waste of bandwidth, and then of energy, is occurring; it then determines how to react to this condition and invokes the hardware management primitives to drive the convenient node operational rate changes. However, the design of the AWM algorithm presented in [16], assumes that the buffer drains at a constant rate, whereas the implementation of the EARTH algorithm in the AWM node, changes the design conditions. For this reason, some minor revisions to the AWM algorithm are necessary to guarantee it works properly.

In particular, changes are related to the value of the β parameter and the estimation of the number of flows. More specifically, in [16] the authors demonstrate that the β parameter determines the speed of convergence of the buffer queue length to the *target* value and its value is calculated by taking into account the output capacity. To meet the designed behavior, as soon as the EARTH algorithm determines any change of operational rate, the value of β is updated by multiplying it by the ratio between the new operational rate value and the old one.

Analogously, the algorithm for the estimation of the number of flows has been designed assuming the buffer service rate does not change: if it does, this change should be taken into account. Therefore, to assure the correct behavior of the AWM algorithm, the estimation of the number of flows works as usual, but the estimated value is multiplied by the ratio between the new operational rate value and the old one.

5. NUMERICAL RESULTS

In this section we will show that the EARTH algorithm,

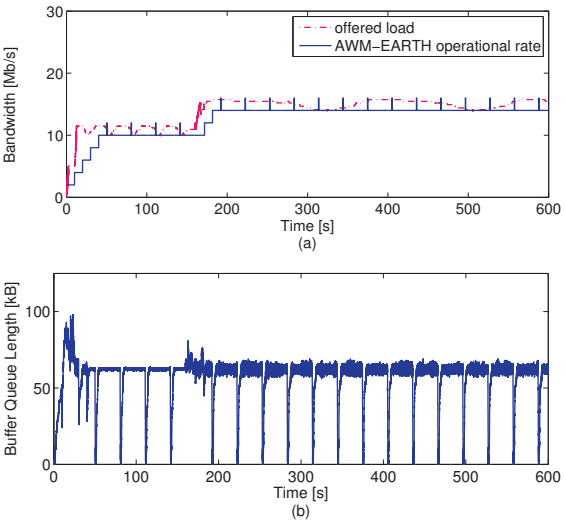

Figure 11: Case Study 1. (a) G-router output capacity determined by the EARTH algorithm - (b) Output Buffer Queue length in the G-router.

combined with the AWM characteristics, is able to determine the node service rate such that it is the minimum between the source offered load and the forward bottleneck capacity along the path. More specifically, AWM stabilizes the buffer queue length around the target when the node is the bottleneck along the path between sources and destinations, so allowing the EARTH algorithm to interpret the condition of empty buffer as a signal that the current node is wasting capacity and consequently power.

We have conducted extensive simulations with the ns-2.30 simulator [26], which we integrated with AWM [27] and EARTH modules. The network topology used is the one depicted in Fig. 10. The considered scenario is typical of home users accessing Internet via DSL technology and sending data to users located anywhere in the Internet. Therefore, for our simulations we consider the source down-link rate is 20Mb/s and the up-link rate is 1Mb/s. The number of source nodes is variable during each simulation and the applications running on each node can generate both FTP-like and Web-like traffic, so assuring the traffic is highly dynamic. In particular we modeled the web workload alternating requests for web files and idle times, with distribution and correlation properties of real web users [28]. More specifically, we assume that short flows arrive according to a Poisson process with an average of 5 new web file requests per second, with Pareto-distributed file sizes with an average of 200 packets and shape 1.35. The average number of FTP-like sources is 10. The round-trip propagation delay is 100ms. The MTU length is set to 1000 bytes. Sources use unmodified TCP NewReno. Let us observe that although we consider TCP NewReno, AWM-EARTH performance are not affected by the TCP version we use; this is because AWM by-passes the TCP control mechanism, which is the key element making the difference between TCP versions. The buffer size is set to 125 packets. The β and *target* values for the AWM algorithm are set to 20kB/s and half of the buffer size, respectively. The *min_thresh* and *max_thresh* parameters for the EARTH algorithm are set to 0 and the average

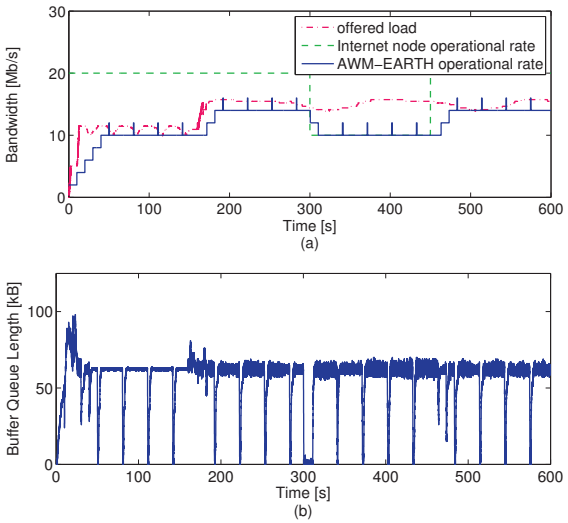

Figure 12: Case Study 2. (a) G-router output capacity determined by the EARTH algorithm - (b) Output Buffer Queue length in the G-router.

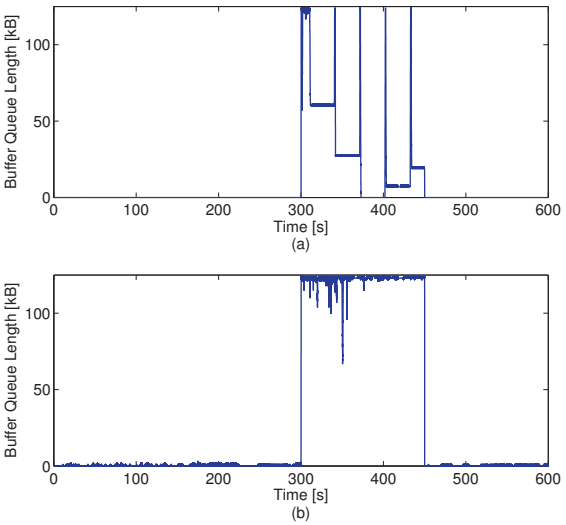

Figure 13: Case Study 2. Output Buffer Queue Length in the Internet node when the access node (a) implements the AWM-EARTH mechanism; (b) is Not Controlled (NC).

between the *target* value e the buffer size, respectively. The time intervals Δt_E, Δt_O and Δt_{P_S} are all set to 10 seconds, whereas Δt_P is set to 30 seconds. The value of ΔC is set to 2Mb/s.

Let us first investigate performance of the AWM-EARTH mechanism in the case the offered load is lower than the network capacity (Case Study 1); for this reason both the maximum available access link capacity and the capacity of any link in Internet are initially large enough to be considered unlimited. Simulation results are shown in Fig. 11.a. The dotted line shows the throughput achieved by the source aggregate in over-provision condition that is when the network capacity is higher than the offered load. The bandwidth demand is about 10Mb/s for the first 200 seconds of simulation and then changes to about 15Mb/s. The solid line is the service rate forced by the EARTH algorithm implemented in the access node. During the first 40 seconds, the EARTH algorithm increases step by step the service rate until it reaches the offered load. As soon as this happens, new attempts to increase the service rate are done every Δt_P seconds but they immediately fails and the service rate steps down again. This event recurs until the offered load changes, after about 200 seconds of simulation: at that time the algorithm detects the offered load change and increases the service rate again until it reaches the offered load.

Fig. 11.b shows the queue length in the access node output buffer. As can be seen, every time the EARTH algorithm increases the service rate more than is necessary, the buffer empties determining the failure of the last attempt. As soon as the service rate steps down again, AWM stabilizes again the buffer queue length around the *target* value. Moreover let us note the buffer never overflows. Simulation results presented so far demonstrate that the AWM-EARTH algorithm is able to follow the source bandwidth demand.

Let us suppose now changes in the traffic loading a forward node occur and make that node a bottleneck along the path between sources and destinations (Case Study 2). In such a conditions, any value of the access node service rate higher than the available bandwidth on the bottleneck node results in a waste of resource. Fig. 12 shows the simulation results obtained in the case a generic node in the Internet path (called "Internet node" in the following) becomes a bottleneck after 300 seconds of simulation. The Internet node bottleneck condition finishes 150 seconds later, that is, after 450 seconds of simulation. In order to demonstrate that AWM-EARTH always determines the minimum service rate between the source offered load and the bottleneck capacity, we expect that during the first 300 seconds of simulation the service rate set by the AWM-EARTH algorithm follows the offered load; then for the following 150 seconds, corresponding to the presence of a bottleneck along the path, it follows the bottleneck capacity; finally it stabilizes again around the offered load.

In Fig. 12.a the service rate determined by the AWM-EARTH algorithm in the access node (solid line) is compared with both the offered load (dash-dotted line) and the Internet node capacity (dashed line): it is evident that the AWM-EARTH mechanism captures the requirements of adapting the access node service rate as expected. Fig. 12.b shows the queue length in the access node output buffer. Again, every time the AWM-EARTH mechanism increases the service rate more than is necessary, the buffer empties determining the failure of the last attempt. As soon as the service rate steps down again, AWM stabilizes the buffer queue length around the *target* value.

To give more insight, Fig. 13 shows the output buffer queue length of the Internet node when the access node implements the AWM-EARTH algorithm (Fig. 13.a) or when it does not implement any energy-aware service rate control mechanism (Fig. 13.b). In the latter case, as soon as the Internet node available bandwidth decreases, its output buffer overflows and remains full until its service rate becomes higher than the offered load. Therefore, for all the duration of this time interval many packets are lost. On the contrary, when the access node implements AWM-EARTH mechanism, the Internet node output buffer remains full and

causes losses for the very short period the AWM-EARTH mechanism needs for capacity adaptation. The above considerations are confirmed by simulation results: when AWM-EARTH is implemented, the number of lost packets is about 80% lower than the not controlled case.

6. CONCLUSIONS

In this paper we have proposed AWM-EARTH, a new mechanism that provides for Active Capacity Scaling capability the access nodes of an ISP network. Results obtained over a realistic ISP topology show that the capacity scaling technique can save up to 70% of total access network power consumption during off peak hours, i.e. more than 45% of total network power consumption. Moreover, the performed simulations have demonstrated that AWM-EARTH is able to adapt the capacity in order to meet the minimum value between the offered load and the forward bottleneck capacity, thus limiting the waste of energy.

We recognize that this work is a first step towards a comprehensive approach. As a future work, we first plan to better assess the performance of the AWM-EARTH mechanism considering different classes of users and QoS constraints. Moreover, we need to quantify the energy that can be saved with this approach. Secondly, we are interested to study a possible real implementation of the proposed algorithm, by exploiting the technology present in modern PCs and mobile devices. Finally, we will explore the possibility of extending the AWM-EARTH mechanism also to the backbone part of the network to further increase the ISP power saving.

7. ACKNOWLEDGMENTS

This work was partially supported by IBM through the PhD. Fellowship Award, by the Italian Ministry of University and Research through the PRIN project EFFICIENT and by the Italian MIUR PRIN project "Sorpasso".

8. REFERENCES

[1] A.P. Sokolov et al., "Probabilistic Forecast for 21st Century Climate Based on Uncertainties in Emissions (without Policy) and Climate Parameters," *MIT Global Change Program Report*, Cambridge, USA, January 2009.

[2] "An inefficient truth," *Global Action Plan Report*, http://www.globalactionplan.org.uk/, December 2007.

[3] M. Pickavet et al., "Worldwide Energy Needs for ICT: the Rise of Power-Aware Networking," *Proc. of IEEE ANTS Conference*, Bombay, India, December 2008.

[4] SMART 2020 Report, *Enabling the low carbon economy in the information age*, http://www.theclimategroup.org, 2008.

[5] S.Pileri, *Energy and Communication: engine of the human progress*, INTELEC 2007 keynote, Rome, Sep. 2007.

[6] M. Etoh et al., "Energy Consumption Issues on Mobile Network Systems," *Proc. of IEEE SAINT 2008*, Turku, Finland, July 2008.

[7] R. Tucker, J. Baliga, R. Ayre, K. Hinton, W. Sorin, *Energy Consumption in IP Networks*, ECOC Symposium on Green ICT 2008, Belgium.

[8] M. Gupta, S. Singh, "Greening of the Internet," *Proc. of ACM SIGCOMM*, Karlsruhe, Germany, August 2003.

[9] P. Barford, J. Chabarek, C. Estan, J. Sommers, D. Tsiang, S. Wright, "Power Awareness in Network Design and Routing," *Proc. of IEEE INFOCOM 2008*, Phoenix, USA, April 2008.

[10] L. Chiaraviglio, M. Mellia, F. Neri, "Reducing Power Consumption in Backbone Networks," *Proc. of IEEE International Conference on Communications (ICC 2009)*, Dresden, Germany.

[11] L. Chiaraviglio, M. Mellia, F. Neri, "Energy-Aware Backbone Networks: a Case-Study," *Proc. of First International Workshop on Green Communications (GreenComm'09)*, Dresden, Germany.

[12] V. Jacobson, "Congestion Avoidance and Control," *Proc. ACM SIGCOMM '88*, 1988, Stanford, USA, August 1988.

[13] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol.1, no. 4, pp. 397-413, August 1993.

[14] S. Floyd, "TCP and Explicit Congestion Notification," *ACM Computer Communication Review*, Vol. 24 No. 5, pages 10-23, October 1994.

[15] D. Katabi, M. Handley and C. Rohrs, "Congestion Control for High Bandwidth-Delay Product Networks," *Proc. of ACM Sigcomm 2002*, Pittsburgh, USA, August 2002.

[16] M. Barbera, A. Lombardo, C. Panarello, G. Schembra, "Active Window Management: an efficient gateway mechanism for TCP traffic control," *Proc. of IEEE ICC 2007*, Glasgow, UK, June 2007.

[17] M. Barbera, M. Gerla, A. Lombardo, C. Panarello, M. Y. Sanadidi, G. Schembra, "Active Window Management: Performance Assessment through an Extensive Comparison with XCP," *Proc. of IFIP Networking 2008*, Singapore, May 2008.

[18] K. Christensen, C. Gunaratne, B. Nordman, "Managing Energy Consumption Costs in Desktop PCs and LAN Switches with Proxying, Split TCP Connections, and Scaling of Link Speed," *International Journal of Network Management*, Vol. 15, No. 5, pp. 297-310, September 2005.

[19] J. Restrepo, C. Gruber, C. Machoca, "Energy Profile Aware Routing," *Proc. of First International Workshop on Green Communications (GreenComm'09)*, Dresden, Germany.

[20] *Cisco Energy Wise* , http://www.cisco.com/.

[21] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, D. Wetherall, "Reducing Network Energy Consumption via Sleeping and Rate-Adaptation," *Proc. of USENIX/ACM NSDI 08*, San Francisco, USA, April 2008.

[22] G. Ananthanarayanan, R. H. Katz, "Greening the Switch," *Proc. of Workshop on Power Aware Computing and Systems (HotPower '08)*, San Diego, USA, December 2008.

[23] R. Bolla, R. Bruschi, A. Ranieri, "Green Support for PC-based Software Router: Performance Evaluation and Modeling," *Proc. of the IEEE International Conference on Communications (ICC 2009)*, Dresden, Germany, June 2009.

[24] P. Mahadevan, P. Sharma, S. Banerjee and P. Ranganathan, "A Power Benchmarking Framework for Network Devices ," *Proc. of NETWORKING 2009*, Aachen, Germany, May 2009.

[25] R. Braden, "Requirements for Internet Hosts - Communication Layers," *RFC 1122*, October 1989.

[26] The network simulator ns-2. http://www.isi.edu/nsnam/ns.

[27] http://www.diit.unict.it/arti/Tools/awm_ns2.zip.

[28] P. Barford, M. Crovella, "Generating representative Web workloads for network and server performance evaluation," *Proc. of ACM SIGMETRICS 1998*, Madison, USA, June 1998.